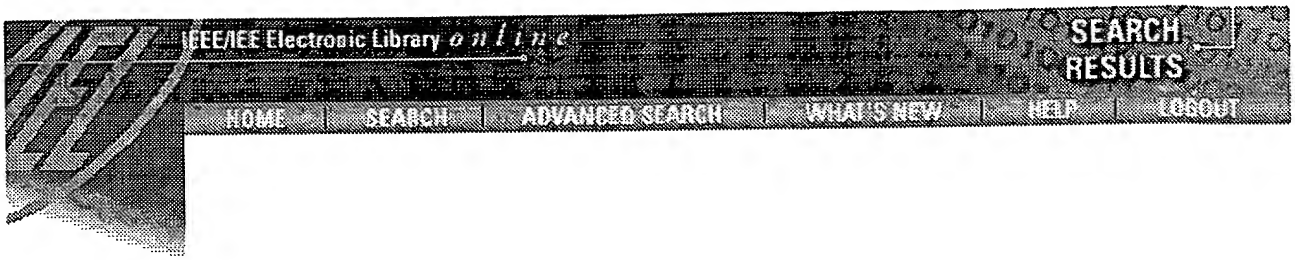

data backup Refine

Collection: ☑ Journals ☑ Conferences ☑ Standards

Your search matched **8** of **656543** documents.

8 are presented on this page, sorted by Score in descending order.

DOC TYPE	VIEW ISSUE TOC	VIEW FULL PAGE	VIEW CITATION
CNF			**Configuring and tuning archival storage systems** *Moore, R.; Lopez, J.; Lofton, C.; Schroeder, W.; Kremenek, G.; Gleicher, M.* Mass Storage Systems, 1999. 16th IEEE Symposium on , 1999 , Page(s): 158 -168
CNF			**Necessity and realization of universally verifiable secret sharing** *Wenbo Mao* Security and Privacy, 1998. Proceedings. 1998 IEEE Symposium on , 1998 , Page(s): 208 -214
CNF			**A linear time, constant space differencing algorithm** *Burns, R.C.; Long, D.D.E.* Performance, Computing, and Communications Conference, 1997. IPCCC 1997., IEEE International , 1997 , Page(s): 429 -436
CNF			**Knowledge engineering approach to data centres disaster backup/recovery planning** *To, C.K.; Ma, V.; Lui, N.* System Sciences, 1989. Vol.III: Decision Support and Knowledge Based Systems Track, Proceedings of the Twenty-Second Annual Hawaii International Conference on , 1989 , Page(s): 241 -248 vol.3
CNF			**Software fault tolerance in a clustered architecture: techniques and reliability modeling** *Lyu, M.R.; Mendiratta, V.B.* Aerospace Conference, 1999. Proceedings. 1999 IEEE Volume: 5 , 1999 , Page(s): 141 -150 vol.5
CNF			**An implementation and performance analysis of backup system using concurrent log processing in real-time DBMS** *Mikyong Han; Yongik Yoon* Real-Time Computing Systems and Applications, 1997. Proceedings., Fourth International Workshop on , 1997 ,

Page(s): 118 -125

CNF  

Backup and storage management in distributed heterogeneous environments

Eitel, P.

Mass Storage Systems, 1994. 'Towards Distributed Storage and Data Management Systems.' First International Symposium. Proceedings., Thirteenth IEEE Symposium on , 1994 , Page(s): 124 -126

CNF  

Safety first (data security)

Greene, R.E.

Information Technology, 1990. 'Next Decade in Information Technology', Proceedings of the 5th Jerusalem Conference on (Cat. No.90TH0326-9) , 1990 , Page(s): 593 -595

| IEL Online Home | Search | Advanced Search | What's New | Help | Logout |
| FAQ's | Support | Comments |

THE HADES FILE SERVER

Hartmut Reuter

Max-Planck-Institut für Plasmaphysik (IPP)
Garching, Germany

ABSTRACT

HADES is a file server for the IBM/370 world under the operating systems VM or MVS. It may be used from CMS, MVS batch, and TSO directly and from UNIX systems via FTP and NFS. HADES has its own data management system independent of the host operating system. It uses automatic data migration onto tape to save disk space and keeps two tape copies of each file. Since there are nearly no limitations in file size and number of files, HADES files can be used easily as bitfiles. Some of the requirements defined in the IEEE Reference Model are fulfilled by an automatic backup process for CMS minidisks that was designed on top of the HADES file system.

DEVELOPMENT HISTORY

HADES is a rather old file server, the development of which started under the name AMOS in 1970 at the Max-Planck-Institut für Plasmaphysik in Garching. At this time, it was designed as an editing system to replace punched cards. It first ran under the IBM operating system MVT as a 'started task', then later under VM as a separate operating system. In 1976, it was brought to the University of Heidelberg and adapted to MVS. In 1981, it came back (together with myself) to Garching under the name Heidelberg Automatic Datamanagement and Editor System (HADES). In 1984, the computing center decided to remove MVS. Therefore, HADES was transformed back to an operating system running under VM on IBM/370 systems. Except for some small assembler parts, it is written completely in PL360 which combines the efficiency of assembler code with the flow control of high-level languages. The kernel of the HADES file server is a multiuser operating system. It has all the nice subtasking and synchronization associated with the 'big' operating systems, but without their expensive protection mechanisms because only trusted code runs in HADES and no user- written programs. The terminal user may only edit his files and call some data management commands.

While the file system originally was designed to keep source code and other small files, it became necessary in the mid-eighties to store large amounts of data originating from experiments in the area of plasma and extraterrestric physics. For this purpose, the file system has been opened to allow very large files (up to 25 GB*) and to remove any restriction for the number of files a user may have.

By now, two kinds of files coexist in HADES. The small files are used to keep program source and job output. These files are subdivided in segments, so the file looks more like a UNIX directory, while the segments correspond to the files in UNIX. The big files are implemented as a sequence of these basic files, but may contain only a single segment. The maximum size is restricted also by the total disk space in a HADES system. At present, there are no 25-GB files because none of the actual running HADES systems have that much disk space. There is also a quota mechanism to prevent users from excessive space allocation.

These big files can easily be used for any kind of file server implementation. They have one additional feature over normal bitfiles, i.e., they may consist of a number of records of undefined size. These records may be addressed directly when reading a file. This offers an easy way to keep a large number of files of a foreign file system in a single HADES file in different records. Such files were used in 1986 to implement an automatic backup system of CMS minidisks and in 1987 to implement an archive system for files on a CRAY computer running COS.

In 1989, HADES got an interface to TCP/IP. An FTP and NFS server have been implemented on the basis of this interface. There are, of course, some restrictions because of the different naming conventions. The HADES file system has also a tree structure like UNIX, but the name of a file or a segment may not exceed ten characters and the names are not case sensitive. Another problem results from the different character representation in the IBM/370 world (EBCDIC instead of ASCII) that makes a conversion necessary if data should be accessible from both worlds. The big HADES files are used by NFS and FTP to archive data in the client's format. They are typically used as backup files for UNIX or PC file systems.

---

* In this paper, KB means 2^{10} bytes, MB means 2^{10} KB, and GB means 2^{10} MB.

CH3039-5/91/0000/0113/$01.00 © 1991 IEEE

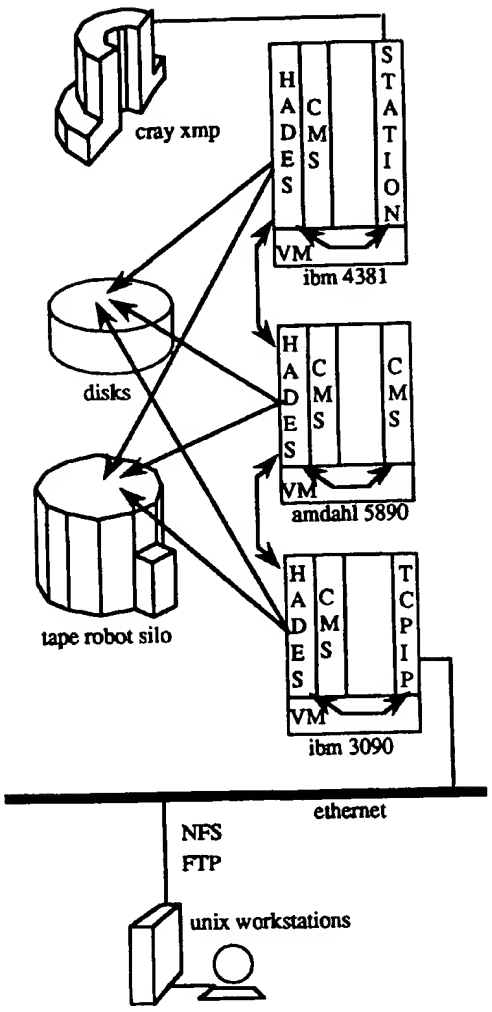

Figure 1. HADES configuration at Garching.

GENERAL DESIGN GOALS

HADES and AMOS always had their own data management system that was completely independent of any host system. It is also independent of the actual disk hardware models since a mapping of virtual disks to real drives take place. From the very beginning a storage hierarchy had been implemented. In AMOS, the secondary medium had changed over time from magnetic tape over the IBM Data Cell to a CDC 38xxx Mass Storage system. Previously, HADES only used magnetic tapes. Today, cartridges in a robot silo are used. The basis for this storage hierarchy is a fast hash code catalogue that contains for each file information about the two obligatory tape copies and the last usage of the file. This allowed the system to use the tape copies that primarily were made to fulfill the data integrity requirements for the migration of files. Each file, which has two current tape copies, may be migrated by just freeing the disk space. The migration itself is controlled by the amount of free disk space, the last usage date, and the size of a file. Today, only about two percent of the data stored in HADES can be kept online. An unused file remains on disk about three days. If a migrated file is going to be used, the fetch from tape is automatically started. Depending on whether the request came interactively or from batch, the requestor gets a message or waits for completion of the fetch processing. HADES owns two tape pools of some thousand tapes. Each file has a tape copy in each of the tape pools that may be kept physically separated to prevent data loss even in the case of fire. A sophisticated tape manager moves files from sparsely filled tapes to others to keep a certain number of tapes free for the next save run. It also controls tapes that are copied after a specified number of years to insure readability even of very old files. All this works perfectly automatically without any human intervention.

RELATIONS TO THE IEEE REFERENCE MODEL

At present, the HADES file system is used by many applications directly associated with the original HADES file names. For these applications, of course, the IEEE reference model cannot be applied. The only two applications running in a mode similar to that described by the reference model are the backup of CMS minidisks and the archiving of CRAY files. The naming conventions and file characteristics in CMS and on the CRAY did not allow the direct use of HADES file names. To reduce the number of files by a factor of about ten to one hundred, many CMS files are stored together into one bitfile using the special property of the HADES bitfiles to contain records of arbitrary length.

In the case of the CMS minidisk backup, a separate process running in the HADES system acts as application client. This process could also run completely outside of HADES and in fact did so the first time. It was incorporated only to avoid data transfer between this process and HADES. The information about the HADES files to be used as bitfiles and the mapping of CMS file names to the records of these bitfiles is contained in a special named HADES file which is allocated automatically for each CMS user when the first backup of his minidisk occurs. This 'index' file is used also by the software running in the CMS machine when the user restores CMS files from HADES or when he displays a list of his archived files at the terminal.

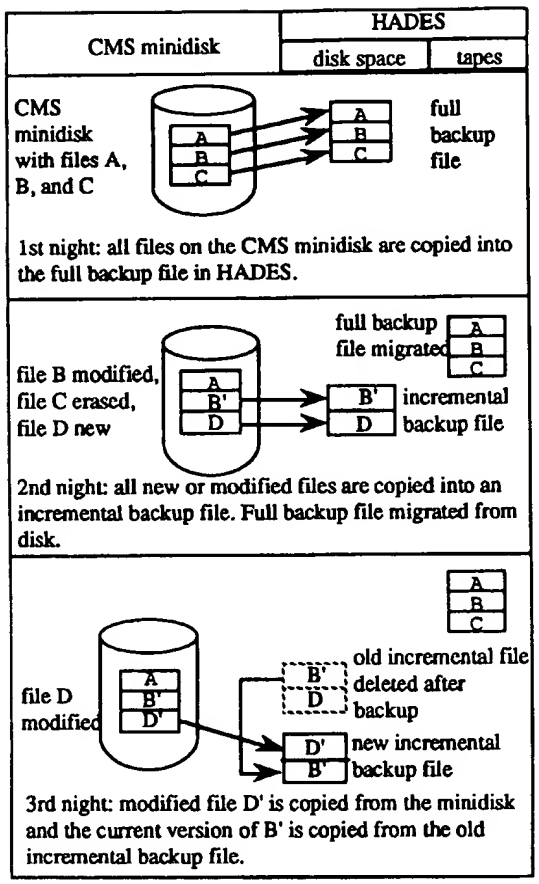

Figure 2. Backup of CMS minidisks (schematic view).

In the case of the CRAY files, the application client was implemented in the CRAY-VM station. It works basically in the same way with a special named index file in HADES. Since the CRAY is running under COS, which does not allow interactive access, the user maintains his archive of CRAY files in the CMS environment in a way similar to the CMS files stored in HADES. At present, the access from UNIX to HADES via FTP and NFS is related to the original HADES file names. But to remove naming restrictions, I am working on a similar application client for NFS and FTP as that which currently exists for CMS and CRAY files.

SECURITY

Each HADES file has an owner and by default only the owner gets access to the file. However, he may grant access explicitly to other users by userid or by group name (in the VM CP sense). The access may be given on different levels such as read, write, and delete.

For CMS users and CRAY jobs under COS, a special authentication is not necessary because the VM interface and the CRAY VM station supply the requestor's userid after the user has passed authentication in these environments. For FTP and NFS, the user must identify himself by userid and password. We are planning to run KERBEROS to avoid the open transfer of passwords over the network.

DATA INTEGRITY

The HADES file system is very stable. Unlike UNIX, HADES has a file locking mechanism to avoid concurrent read and write access to a file. So the only loss of data may be due to hardware failures.

The system is designed to be restartable even after the loss of all disks, and can be restarted from tape. For this purpose, the actual state of the catalogue, the tape volume tables of contents, and other system information are checkpointed twice a day onto tape. If the disks are lost due to a head crash or whatever, HADES has to be restarted with the new disks with some special start parameter and then will guide the operator by a dialog to format the new disks and to reload all system information from the last checkpoint tape. The formatting may take some time, but the reloading of the system is a task of only a few minutes. After that, HADES is usable again. However, all files are migrated and must be restored from tape. Fortunately, we never had such a loss. However, to get some practice, I once simulated this case when it became necessary to move HADES to some other disk drives. It worked perfectly, only that the queues for fetching files from tape got some 700 entries and the users had to wait longer than normal.

SCALABILITY

The actual configuration of the system is read in at start time. The maximum number of files, tapes, and users a given configuration can support depends on the disk space available to the system. So it is possible to run a very small test system with only some MB of disk space with the same HADES module that runs in a production environment with 10 GB disk space or more. Presently, IBM 3380, 3370, and 3350 disks are supported by HADES. But new models can be supported very easily because a mapping of virtual to real disks by means of some tables is used to keep the system independent of real hardware. Only the very lowest driver level knows about the real disk models. There is no design limit for the disk space up to about 100 GB and 30 thousand tapes. However, each addition of disk space requires a reorganization of the hash code-based catalogue that cannot be done during the normal file server operation.

At Garching, HADES has presently about 10 GB of disk space and 2,500 tapes. It keeps 62,000 files with a total size of about 240 GB. The medium file size of nearly 4 MB is relatively high because of the multi-record structure of the bitfiles mentioned above. So the number of CMS or CRAY files stored in HADES is much higher.

Sites with more than one VM or MVS system may run multiple HADES systems in a cluster that shares the disks and tape pools. The HADES systems are then coupled typically by channel-to-channel adapters to allow the exchange of locking information. The coupling is done by a master slave protocol where the master system runs nearly without performance degradation while the slave systems have to acquire locks for all sensitive actions. The data rates, which are reached in the slave systems, are only slightly lower than in the master system and much higher than it would be to transfer all the data over a network or a channel to a channel adapter. At Garching, three VM systems are coupled: a VM system with MVS at Heidelberg, and two VM systems at Cornell University.

DATA RATES AND DATA TRANSFER

The connection to the LAN is realized by our own version of socket interfaces that communicate with the IBM-supplied TCP/IP software under VM. At Garching, the IBM TCP/IP software is connected to Ethernet, but it could also give access to faster networks such as FDDI or HIPPI. In comparison to systems based on newer hardware, such as disk farms and fast networks, the data rates reached by HADES are low. But in the CMS environment, data rates of 0.7 to 1 MB/second are not too bad. The rates reached on the Ethernet have been improved by version 2 of the IBM TCP/IP software to about 140 to 170 KB/second.

About 550 MB are stored from outside into HADES and twice that (1.1 GB) are read out of HADES per day. The overall internal data transfer from disk to tape (save copies of file) and from tape to disk (fetch migrated files) is about 3 GB each per day, while the transfer from tape to tape is about 5 GB per day. This last value, of course, depends strongly on the medium load of the tapes that presently is rather high. The total internal data transfer is by a factor 7 higher than the external one. This high internal data flow is the price paid for the high data integrity guaranteed by HADES (first tape copy of a file within 12 hours, second within a day) and because of the small online disk space the system may use for staging at our site. It is, however, not very expensive: HADES consumes only about one percent of the CPU power of the Amdahl 5890-200 on which the main HADES system runs.

The annual data growth was about 100 GB during the past two years, or about 270 MB per day. The highest data growth per day we observed was 3.5 GB.

Partner	Data volume MB/day	Data rate KB/s
VM users	1400	700 - 1000
UNIX users	250	140 - 170
internal disk/tape	3000	1000
internal tape/tape	5000	1000

Table 1: External and internal data transfer.

CONCLUSION

HADES does not conform to the rules of the IEEE Reference Model, but it is at least a reliable file server based on the reliable IBM/370 hardware. The big HADES files may be used as bitfiles in future file server implementations that meet the IEEE Reference Model requirements. The integerated data migration makes a separate archiving, as required by some other file servers, unnecessary. Therefore, a rather clear structure would result. The use of PL360 as programming language makes HADES unportable to other instruction sets, but it makes the code very efficient and allows a powerful file server to run on a cheap and reliable machine such as an IBM 4381. With the current hardware architecture, HADES will not be able to play in the high end of the performance range. This would require more specialized hardware that is currently being developed at many places.

Also at Garching, a trend is underway to develop a more uniform UNIX environment (nevertheless, the next vector or parallel computer will run UNIX). Therefore, the network interface will become increasingly important and additional services such as the Andrew File System should be offered. Since some of the experiments at Garching using the CMS environment have a rather long lifetime, HADES will remain the main file server at Garching for at least some years. During this time, the development of new and more standardized interfaces to the UNIX world will continue.

REFERENCES

1. Reuter, Hartmut, and Peter Sandner, "Das Heidelberger Datenverwaltungs-und Editor-Subsystem HADES," *Das Rechenzentrum*, Vol. 1, 1983.

2. Reuter, Hartmut, "HADES - a File Server for the VM Environment," *Proceedings*, MSS Issues and Directions, Ithaca, NY, November 1989.

ACM Digital Library
search results

Page: **1 of 1**
Articles: 1-2 of 2 Ordered By Score

automatic backup

Search: New | Undo | Refine
Order By: Publication | Score | Publication Date
View: Brief Listing | Full Listing | Search Expression | All Articles |
+Page Size | -Page Size | Help

No.	Article	Score
1)	An editor for revision control; Christipher W. Fraser and Eugene W. Myers; *ACM Trans. Program. Lang. Syst.* 9, 2 (Apr. 1987), Pages 277 - 295 [Find Related Articles]	8
2)	Managing APL public code for an in-house APL system (before and after LOGOS); D. F. Stoneburner; *Conference proceedings on APL in transition*, 1987, Pages 210 - 215 [Find Related Articles]	8

go to page: **1 of 1**